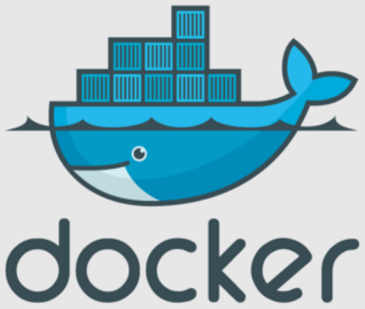
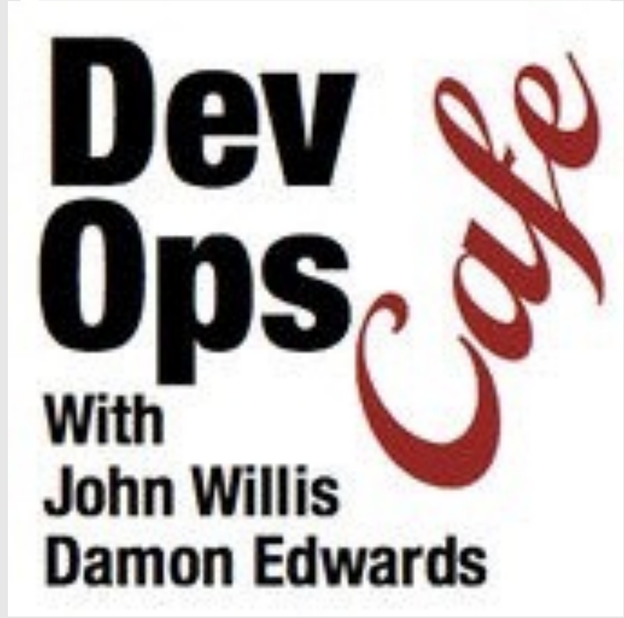
Devops’n the Operating System

John Willis
Director of Ecosystem Development
Docker, Inc.



@botchagalupe

- a.k.a. John Willis
- 35 Years in IT Operations
- Exxon, Canonical, Chef, Enstratius, Socketplane
- Devopsdays Core Organizer
- 35 Official Devopsdays
- Devopscafe on iTunes
- Organizer of Devops Enterprise Summit



Devops

Devops is a movement motivated to turn human capital into high performance organizational capital.

Norton Ghost 2003 Copyright (C) 1998-2002 Symantec Corp. All rights reserved.

Progress Indicator

0% 25% 50% 75% 100%

Statistics

Percent complete	84
Speed (MB/min)	245
MB copied	2246
MB remaining	401
Time elapsed	9:10
Time remaining	1:38

Details

Connection type	Local
Source	**Local drive [1], 16378 MB**
Destination	**Local drive [2], 5114 MB**
Current partition	1/2 Type:83 [Linux], Size: 15680 MB
Current file	/home/alex/local/bin/mono

symantec.

First Generation Configuration Management

Tivoli - Configuration Manager

BMC - Bladelogic

HP - Opsware

```
install_aix_package
                  image_dir = "/tmp/installp"
                  source_dir = "/installp"
                           is_image_remote = n
                           keep_images = y
                           package_file = Adobe
                           log_mode = high
                           log_path = /tmp/installp/log_Acrobat
                           report_log = y
                           block_size = 512
                           override_files = n
                           install_root = n
                           install_share = n
                           install_usr = n
                           cdrom_volume = n
                           install_corequisites = n
                           is_update = n
                           expand_fs = n
                           save_directory = /tmp/installp/salva_Acrobat3.01

                       fileset
                          name = Adobe.acrobat
                                    level = 3.0.1.0
                                    description = "Adobe Acrobat reader for AIX"
                       end

end
```

HJK SOLUTIONS

We Launch Startups.

Why are people paying 3 to 5 million for configuration management software? 3

Posted by Adam Jacob on 8/31/2007

In John Willis' response to Puppet, iLike and Infrastructure 2.0, he poses the question:

> Maybe I should ask this one more time... and why are people paying in excess of 3 to 5 million a y

I think the answer revolves around two things.

People don't realize that manual systems administration is a prob

By "people" I mean everyone involved... many excellent systems administrators just don't see the need fc week building my current systems by hand?")

The result of this is that automation comes in too late in the game, after you've already got a couple of h support structures built around doing it by hand. If you're a growing startup, this winds up impacting yo like mine to help, or at least talking to Luke's own Reductive Labs to help get you running quickly.

If you are the Fortune 1000, it's another thing altogether. You start to look for a way out, and someone t it's hard to be nimble enough to realize that Open Source provides a better path here.

You have to see it to believe it.

If you ask anyone working in technology whether they want the ability to rebuild the entire infrastructure

What follows, though, is "but our infrastructure can't work that way because of X". Or "your systems mus of functionality. They've never even seen it. So they believe you when they say it, but they don't have any

When you realize you must have it, and you already believe that it's impressively difficult to do, having s **believe**.

HJK SOLUTIONS

We Launch Startups.

Puppet, iLike and Infrastructure 2.0 2

Posted by Adam Jacob on 8/31/2007

John Willis, one of the founders of Gulf Breeze Software (an IBM Tivoli consulting house,) met up with Luke Kanies and interviewed him about Puppet.

In addition to lots of insightful commentary on how Puppet is constructed, and a nice compare/contrast with how Tivoli is built (and you would be hard pressed to talk to someone who knows more about how Tivoli is built than John Willis, I expect,) there is also a section about iLike and HJK:

> Uncomfortable with his recent celebrity at conferences, Luke told me that he has difficulty measuring his successes because he has his head so deep in the development and services of Puppet. One of his better success stories is with iLike.com, a website that allows users to download and share music. When iLike created one of the first Facebook applications, it grew from about ½ million users to over 6 million in a week. Luke, being the entrepreneur that he is, asked how iLike planned to manage that growth. He discovered that a services company in Seattle was managing iLike.com's infrastructure build out using Puppet. In fact, one of the owners of that company told Luke that he makes a healthy living installing Puppet. Luke admitted that he felt feel pretty good to know that other people can make a living from his product.

Links

HJK Solutions
Contact Us

Archives

August 2007 (2)
July 2007 (4)
June 2007 (2)

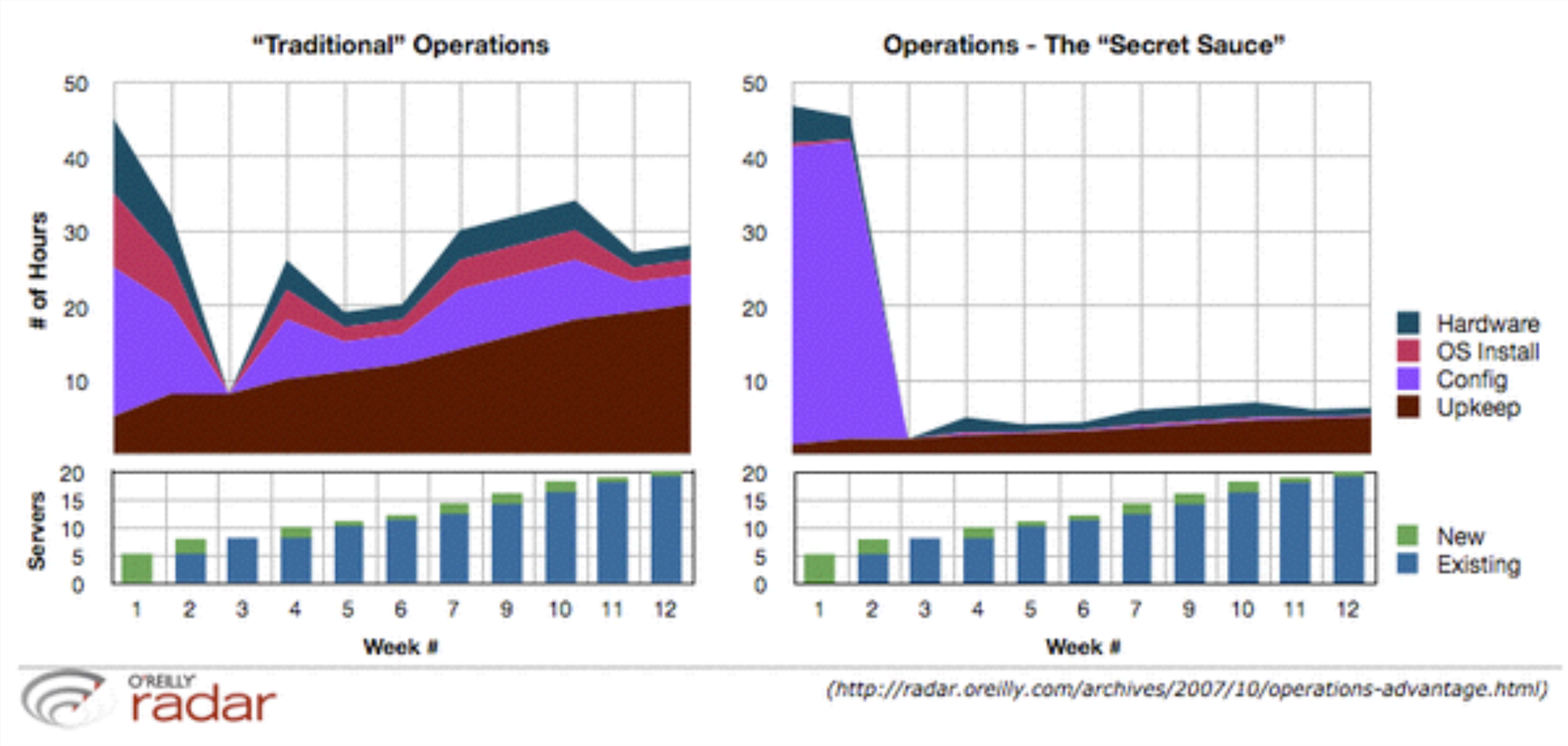
Syndicate

Articles
Comments
Trackbacks

Categories

news (3)
solutions (8)

Operations is a competitive advantage… (Secret Sauce for Startups!)



Second Generation Configuration Management

Cfengine

Puppet

Chef

```
package 'apache2' do
  package_name node['apache']['package']
  default_release node['apache']['default_release'] unless node['apache']['default_release'].nil?
end

%w(sites-available sites-enabled mods-available mods-enabled conf-available conf-enabled).each do |dir|
  directory "#{node['apache']['dir']}/#{dir}" do
    mode '0755'
    owner 'root'
    group node['apache']['root_group']
  end
end

%w(default default.conf 000-default 000-default.conf).each do |site|
  link "#{node['apache']['dir']}/sites-enabled/#{site}" do
    action :delete
    not_if { site == "#{node['apache']['default_site_name']}.conf" && node['apache']['default_site_enabled'] }
  end

  file "#{node['apache']['dir']}/sites-available/#{site}" do
    action :delete
    backup false
    not_if { site == "#{node['apache']['default_site_name']}.conf" && node['apache']['default_site_enabled'] }
  end
end
```

History of Virtualization

- IBM 360/370 (1960/1970)
- CHROOT - Version 7 Unix 1979 (Bell Labs) and BSD in 1982 (Berkley)
- VMware (1998)
- FreeBSD Jails 2000
- XEN 2003
- Solaris Zones 2004
- OpenVZ 2005
- **Amazon Web Services 2006**
- BTRFS (Oracle) 2007
- Namespaces 2007
- Cgroups (Google) 2007
- KVM 2007
- AIX LPARS (IBM) 2007
- Drawbridge (2008)
- Hyper-V (2008)
- Linux Containers - LXC (Parelles, IBM, Google) 2008
- Docker (Dotcloud Inc) 2013
- Rocket (Coreos) 2014
- Unikernels (2015)

VM Image Sprawl in Real Life

Posted on Jan 1, 2009 by Randy Bias

A while back, Geva Perry and I were chatting about the issue of virtual machine image sprawl (Google Search), which is really little more than an extension of not-so-new traditional physical server sprawl problem. It's hard to get really hard data on how bad the vm sprawl problem is since most images exist behind firewalls or other walled gardens. However, there is one good place to get solid data and that's Amazon's own public image repository.

Real Data So, around the time we were chatting I started collecting some information on the number of public Amazon Machine Image (AMIs). This isn't a perfect sampling, but should be pretty good for most purposes.

Web Operations, 1st Edition

Keeping the Data On Time

By: Jesse Robbins, John Allspaw

Released: June 2010

5.0

Read 1 Review

Write a review

Register Your Book

View/Submit Errata

Media Praise

Ask a Question

Bulk Discounts & Licensing

Listen

Chapter 5. Infrastructure As Code

Adam Jacob

YOU'RE SITTING AT HOME, WATCHING A MOVIE AND EATING POPCORN, with your feet up and the family gathered around you. The phone rings—it's your on-call system administrator.

"The datacenter has been hit by a tornado—it ripped right through our cage. What do we do?"

Once you get over the obvious answer,[1] you start running down the list:

1. Pause the movie.
2. Sign up for an account with a cloud computing provider, to replace the raw computing, network, and storage resources you have lost.
3. Start uploading/downloading the off-site backups of your customer and application data to the new infrastructure.
4. Provision enough servers to bring the company back online, assigning an appropriate role to each new server resource ("web server," "database server," "monitoring server," etc.).
5. Change your DNS to point to your new infrastructure, with a "we got hit by a tornado" page.
6. Restore the customer and application data.
7. Remove the "we got hit by a tornado" page.
8. Finish the movie.

NETFLIX

The Net

Saturday, August 13, 2011

Building with Legos

In the six years that I have been involved in building and releasing software here at Netflix, the process has evolved and improved significantly. When I started, we would build a WAR, get it setup and tested on a production host, and then run a script that would stop tomcat on the host being pushed to, rsync the directory structure and then start tomcat again. Each host would be manually pushed to using this process, and even with very few hosts this took quite some time and a lot of human interaction (potential for mistakes).

Our next iteration was an improvement in automation, but not really in architecture. We created a web based

Virtualization

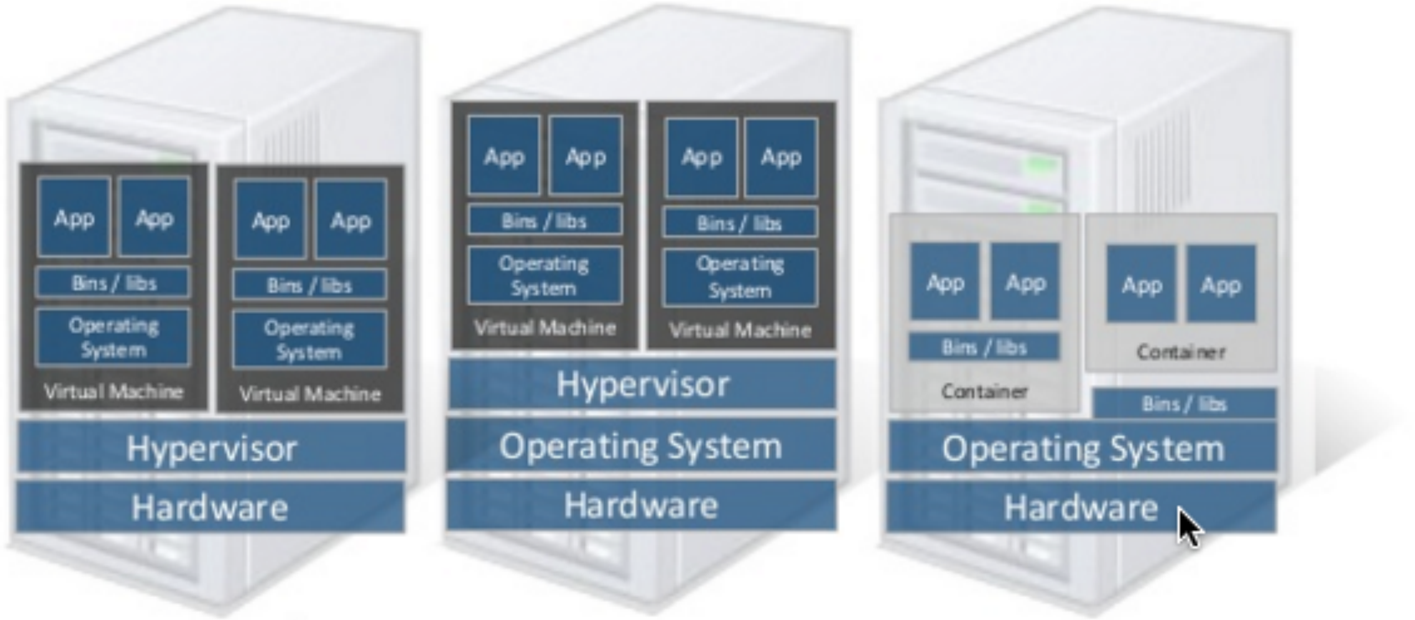
- **Type 1 Virtualization**
  - VMware ESX, XEN, Hyper-V
  - (indirectly Amazon, Rackspace, etc..)
- **Type 2 Virtualization**
  - KVM, Virtualbox, QEMU, VMware Workstation
  - (indirectly Vagrant)
- **OS Level Virtualization**
  - OpenVZ, LXC, Docker

Hypervisors vs. Linux Containers

IBM

Containers share the OS kernel of the host and thus are lightweight.
However, each container must have the same OS kernel.

Containers are isolated, but share OS and, where appropriate, libs / bins.



Type 1 Hypervisor | Type 2 Hypervisor | Linux Containers

http://www.slideshare.net/BodenRussell/realizing-linux-containerslxc

Why OS Level Virtualization

- Provision in milliseconds
- Near bare metal runtime performance
- VM-like agility – it’s still “virtualization”
- Lightweight – Just enough Operating System (JeOS)
- Supported with modern Linux kernel
- Growing in popularity

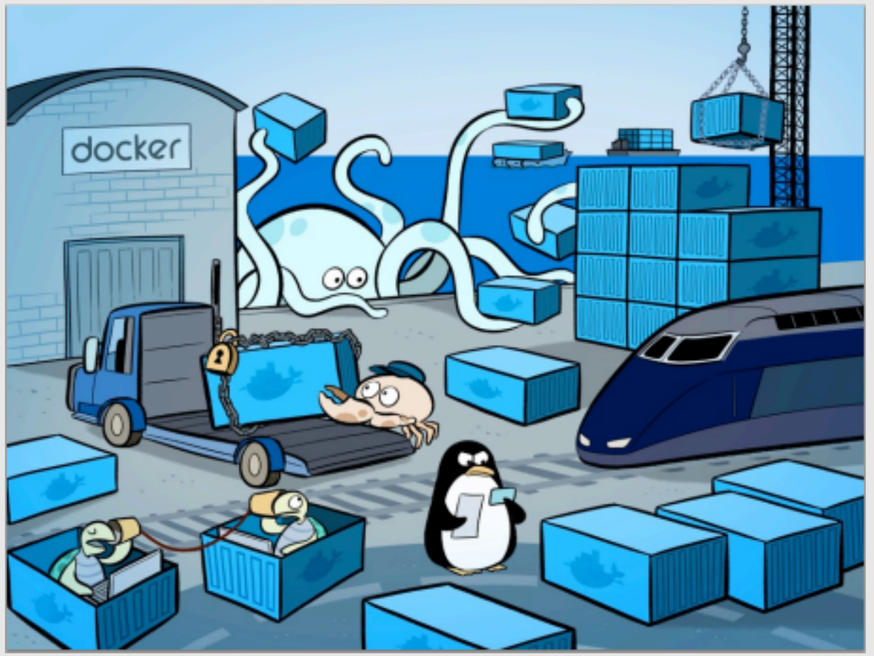
Introducing Containers

Containerization uses the kernel on the host operating system to run multiple root file systems

- Each root file system is called a **container**
- Each container also has its own
  - Processes
  - Memory
  - Devices
  - Network stack

Docker?

- Isolation
- Lightweight
- Simplicity
- Workflow
- Community



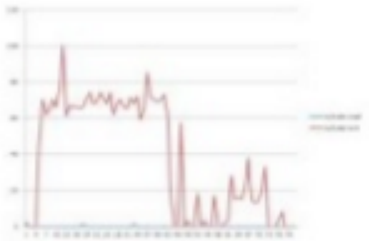
Density & Footprint – Docker

Clip slide

- *In this test, we created 150 Docker containers with CentOS, started apache & then removed them*
- *Average footprint was ~10MB per container*
- *Average start time was 240ms*

- **Serially booting 150 containers which run apache**
  - Takes on average 36 seconds
  - Consumes about 2% of the CPU
  - Negligible HDD space
  - Spawns around 225 processes for create
  - Around 1.5 GB of memory ~ 10 MB per container
  - Expect faster results once docker addresses performance topics in the next few months
- **Serially destroying 150 containers running apache**
  - On average takes 9 seconds
  - We would expect destroy to be faster – likely a docker bug and will triage with the docker community

CPU profile

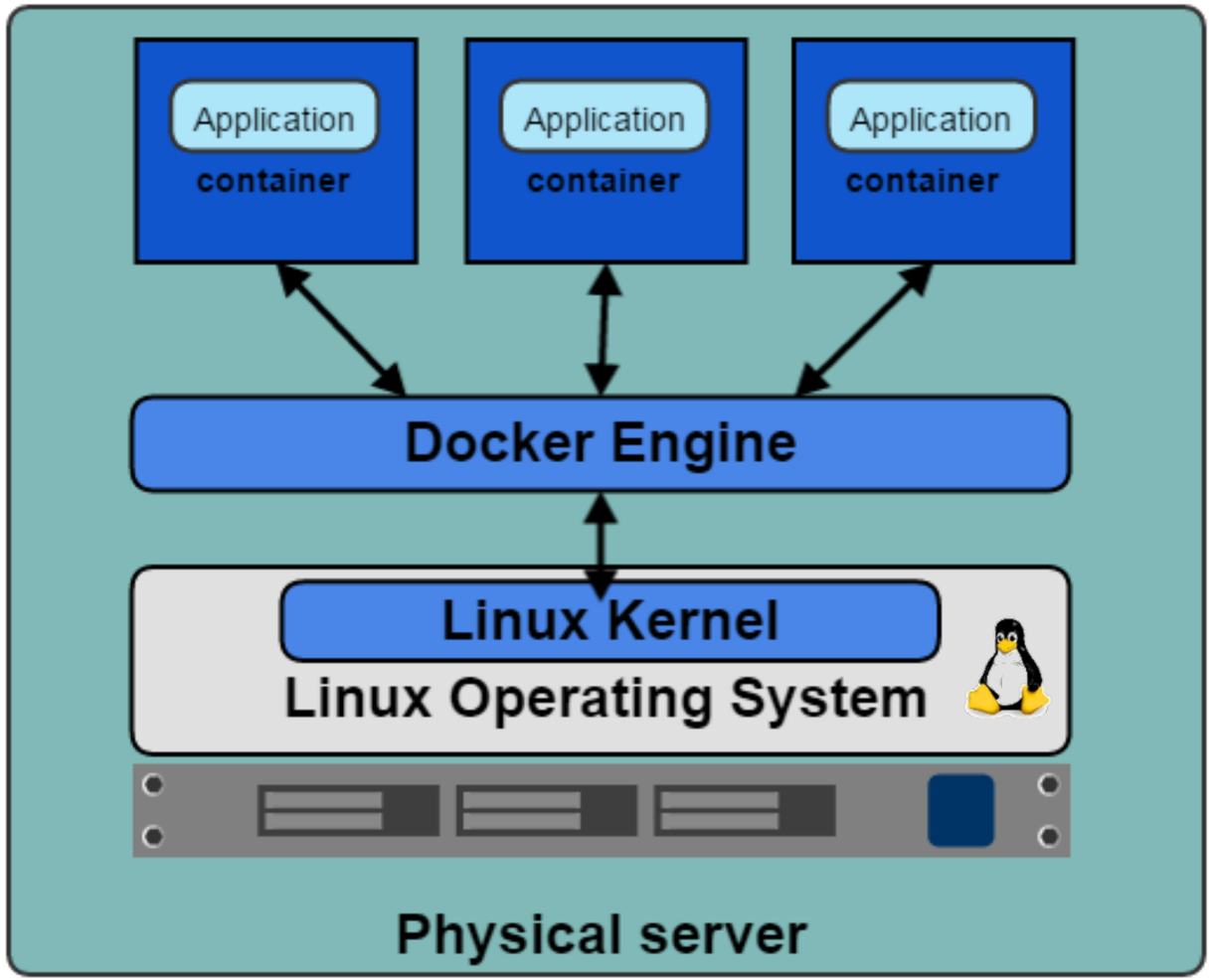


I/O profile

http://www.slideshare.net/BodenRussell/realizing-linux-containerslxc

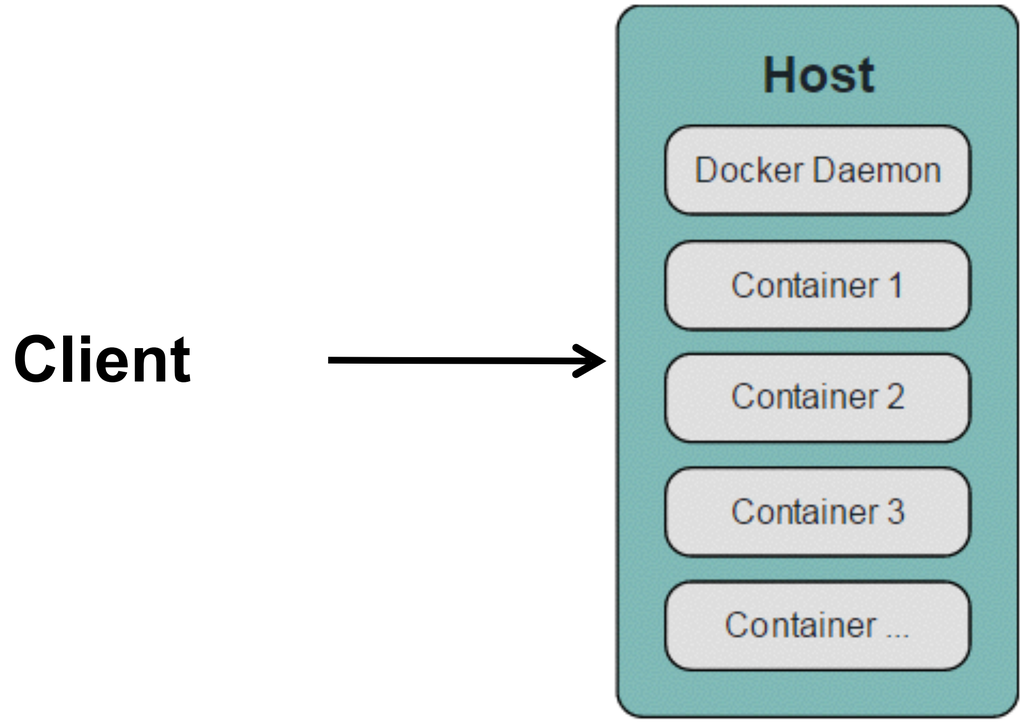
Docker and the Linux Kernel

- **Docker Engine** is the program that enables containers to be distributed and run
- Docker Engine uses Linux Kernel namespaces and control groups
- Namespaces give us the isolated workspace



Docker Client and Daemon

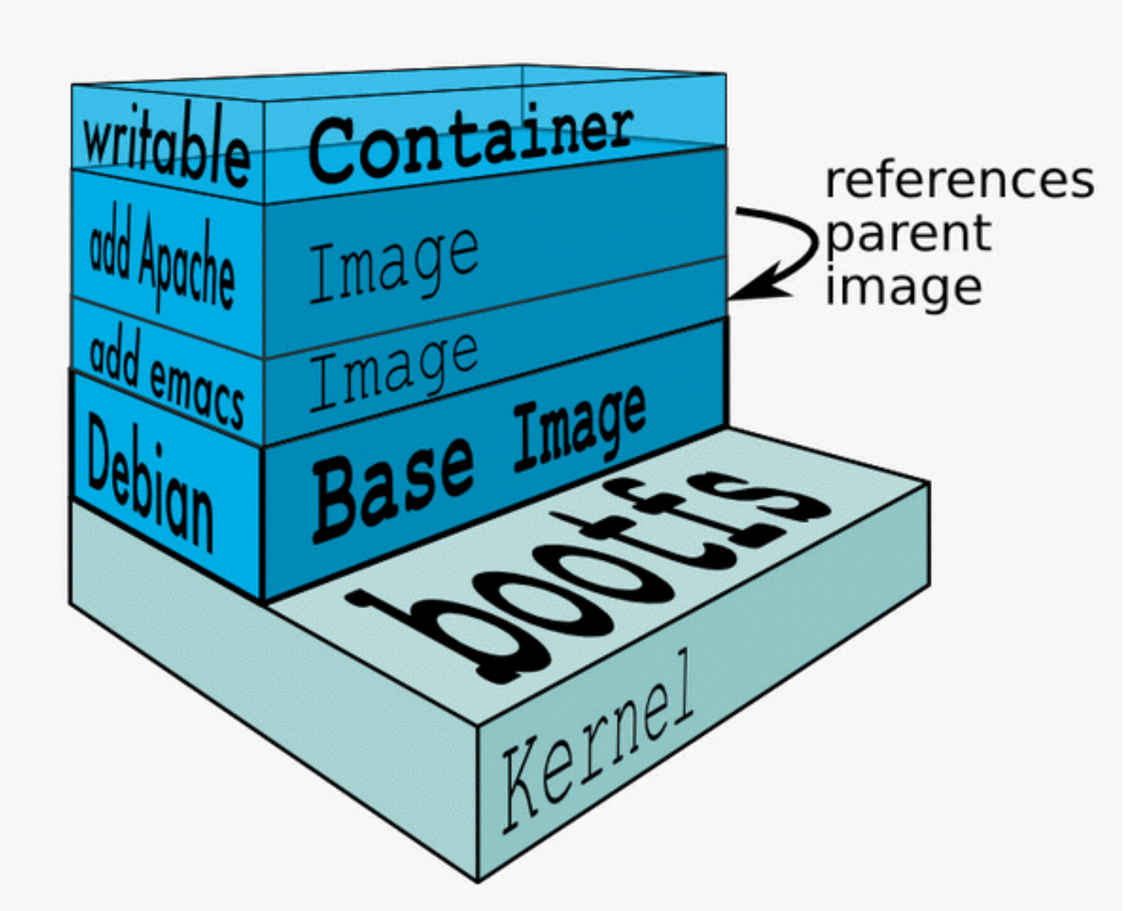
- Client / Server architecture
- Client takes user inputs and sends them to the daemon
- Daemon runs and distributes containers
- Client and daemon can run on the same host or on different hosts
- CLI client and GUI (Kitematic)



Understanding image layers

- An image is a collection of files and some meta data
- Images are comprised of multiple layers
- A layer is also just another image
- Each image contains software you want to run
- Every image contains a base layer
- Docker uses a copy on write system
- Layers are read only

- COW/Union Filesystems (AUFS/BTRFS)



Dockerfile Examples

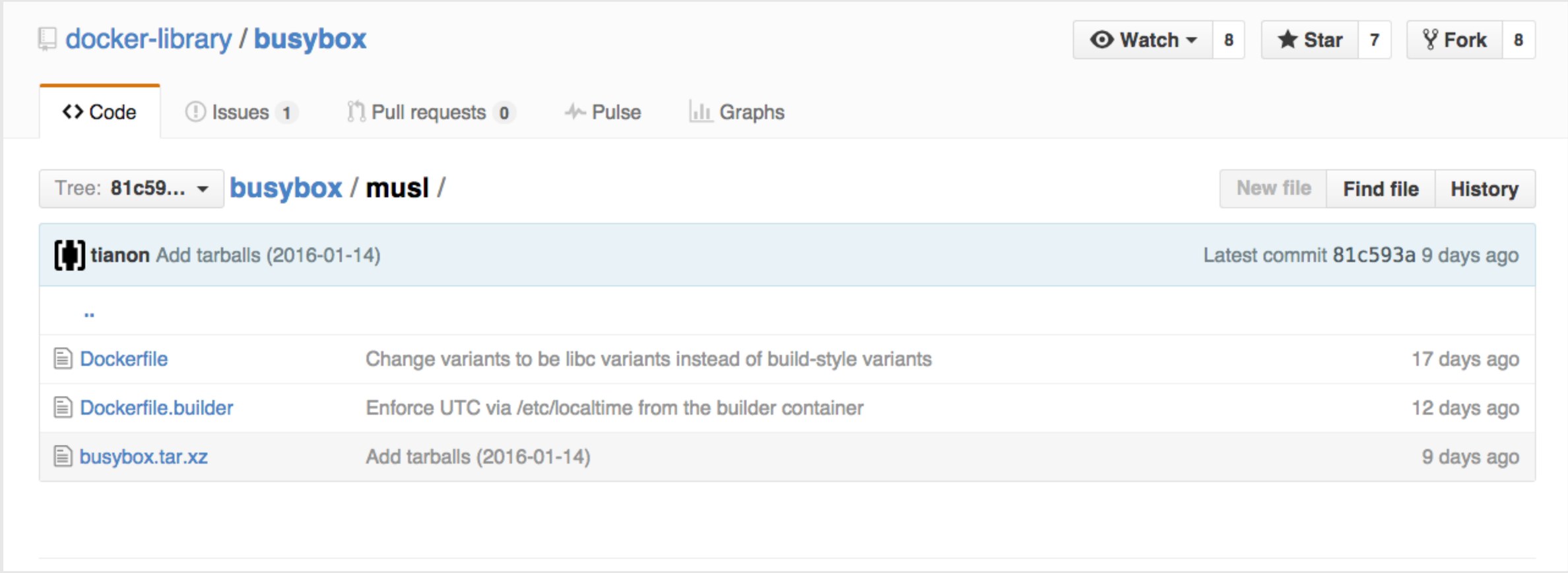
```
FROM scratch
ADD ubuntu-trusty-core-cloudimg-amd64-root.tar.gz /

# a few minor docker-specific tweaks
# see https://github.com/docker/docker/blob/master/contrib/mkimage/debootstrap
RUN echo '#!/bin/sh' > /usr/sbin/policy-rc.d \
        && echo 'exit 101' >> /usr/sbin/policy-rc.d \
        && chmod +x /usr/sbin/policy-rc.d \
        \
        && dpkg-divert --local --rename --add /sbin/initctl \
        && cp -a /usr/sbin/policy-rc.d /sbin/initctl \
        && sed -i 's/^exit.*/exit 0/' /sbin/initctl \
        \
        && echo 'force-unsafe-io' > /etc/dpkg/dpkg.cfg.d/docker-apt-speedup \
        \
        && echo 'DPkg::Post-Invoke { "rm -f /var/cache/apt/archives/*.deb /var/cache/apt/archives/partial/*.deb /var/cache/apt/
        && echo 'APT::Update::Post-Invoke { "rm -f /var/cache/apt/archives/*.deb /var/cache/apt/archives/partial/*.deb /var/cac
        && echo 'Dir::Cache::pkgcache ""; Dir::Cache::srcpkgcache "";' >> /etc/apt/apt.conf.d/docker-clean \
        \
        && echo 'Acquire::Languages "none";' > /etc/apt/apt.conf.d/docker-no-languages \
        \
        && echo 'Acquire::GzipIndexes "true"; Acquire::CompressionTypes::Order:: "gz";' > /etc/apt/apt.conf.d/docker-gzip-index

# enable the universe
RUN sed -i 's/^#\s*\(deb.*universe\)$/\1/g' /etc/apt/sources.list

# overwrite this with 'CMD []' in a dependent Dockerfile
CMD ["/bin/bash"]
```

Dockerfile Examples



Socketplane Example

Branch: master ▾ **docker-ovs / Dockerfile** Find file | Copy path

dave-tucker Add OVS 2.4.0 — fede885 on Nov 27, 2015

1 contributor

32 lines (31 sloc) | 1.32 KB — Raw | Blame | History

```
1  FROM socketplane/busybox:latest
2  MAINTAINER The SocketPlane Team <support@socketplane.io>
3  ENV OVS_VERSION 2.4.0
4  ENV SUPERVISOR_STDOUT_VERSION 0.1.1
5  # Configure supervisord
6  RUN mkdir -p /var/log/supervisor/
7  ADD supervisord.conf /etc/
8  # Install supervisor_stdout
9  WORKDIR /opt
10 RUN mkdir -p /var/log/supervisor/
11 RUN mkdir -p /etc/openvswitch
12 RUN wget https://pypi.python.org/packages/source/s/supervisor-stdout/supervisor-stdout-$SUPERVISOR_STDOUT_VERSION.tar.gz --no-c
13         tar -xzvf supervisor-stdout-0.1.1.tar.gz && \
14         mv supervisor-stdout-$SUPERVISOR_STDOUT_VERSION supervisor-stdout && \
15         rm supervisor-stdout-0.1.1.tar.gz && \
16         cd supervisor-stdout && \
17         python setup.py install -q
18 # Get Open vSwitch
19 WORKDIR /
20 RUN wget https://s3-us-west-2.amazonaws.com/docker-ovs/openvswitch-$OVS_VERSION.tar.gz --no-check-certificate && \
21         tar -xzvf openvswitch-$OVS_VERSION.tar.gz &&\
22         mv openvswitch-$OVS_VERSION openvswitch &&\
23         cp -r openvswitch/* / &&\
24         rm -r openvswitch &&\
25         rm openvswitch-$OVS_VERSION.tar.gz
26 ADD configure-ovs.sh /usr/local/share/openvswitch/
27 # Create the database
28 RUN ovsdb-tool create /etc/openvswitch/conf.db /usr/local/share/openvswitch/vswitch.ovsschema
29 # Put the OVS Python modules on the Python Path
30 RUN cp -r /usr/local/share/openvswitch/python/ovs /usr/lib/python2.7/site-packages/ovs
31 CMD ["/usr/bin/supervisord"]
```

Docker and Windows

- **Azure**
  - Azure Container Service
  - Swarm Integration
- **Windows Server 2016**
  - Windows Server Containers
  - Hyper-V Containers

Immutable Infrastructure

NETFLIX

The Netflix Tech Blog

Saturday, August 13, 2011

Building with Legos

In the six years that I have been involved in building and releasing software I
evolved and improved significantly. When I started, we would build a WAR, g
production host, and then run a script that would stop tomcat on the host bei
structure and then start tomcat again. Each host would be manually pushed
with very few hosts this took quite some time and a lot of human interaction

Our next iteration was an improvement in automation, but not really in archit
tool that would handle the process of stopping and starting things as well as
extracting the new code. This meant that people could push to a number of s
check boxes. The tests to make sure that the servers were back up before p
automated and have failsafes in the tool.

Links

Netflix US & Canada Blog

MARTIN FOWLER

Intro Videos Design Agile Refactoring NoSQL DSL Continuous Delivery Microservices

ImmutableServer

Kief Morris
13 June 2013

Automated configuration tools (such as CFEngine, Puppet, or Chef) allow you to specify how servers should be configured, and bring new and existing machines into compliance. This helps to avoid the problem of fragile SnowflakeServers. Such tools can create PhoenixServers that can be torn down and rebuilt at will. An Immutable Server is the logical conclusion of this approach, a server that once deployed, is never modified, merely replaced with a new updated instance.

@bglpe

Immutable Matters

Why Order Matters: Turing Equivalence in Automated Systems Administration

Steve Traugott - TerraLuna, LLC
Lance Brown - National Institute of Environmental Health Sciences

Pp. 99-120 of the *Proceedings of LISA '02: Sixteenth Systems Administration Conference*, (Berkeley, CA: USENIX Association, 2002).

"The least-cost way to ensure that the behavior of any two hosts will remain completely identical is always to implement the same changes in the same order on both hosts."

Management Methods

- Divergence
- Convergence
- Congruence

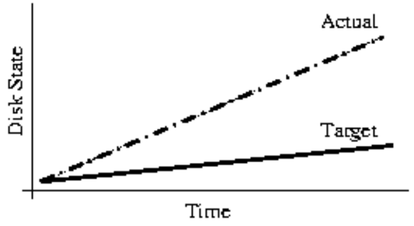


Figure 1: Divergence.

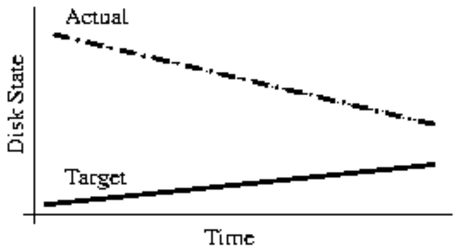


Figure 2: Convergence.

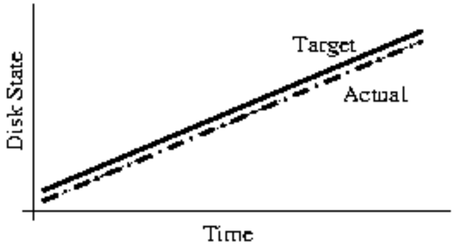


Figure 3: Congruence.

Immutable Delivery

Gartner

WHY GARTNER ANALYSTS RESEARCH EVENTS CONSULTING ABOUT

Assessing Docker and Containers for Five Software Delivery Use Cases

27 April 2015 G00275476

Analyst(s): Richard Watson

Summary

Docker offers application-focused, container-based virtualization to DevOps-minded developers and administrators. This document assesses Docker for use cases spanning development and test, continuous integration (CI), production deployment, and building private PaaS.

Already ha

Sign in to v

Enter Use

Enter Pas

SIGN IN

Forgot use

docker What is Docker? Use Cases Try It! Install & Docs Br

May 26, 2015

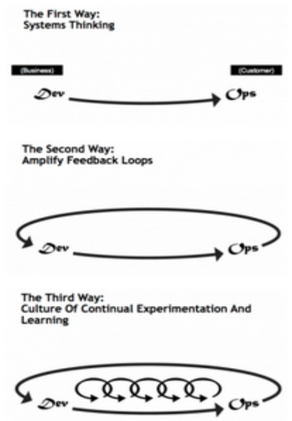
DOCKER AND THE THREE WAYS OF DEVOPS

written by John Willis, Evangelist at Docker

Have you read Gene Kim's *The Phoenix Project*? Some of the principles behind the Phoenix Project and an upcoming book I am co-authoring with Gene (The DevOps Cookbook) have been referred to as the "Three Ways of DevOps". These are particular patterns of applying DevOps principles in a way that yields high performance outcomes.

We assert that the Three Ways describe the values and philosophies that frame the processes, procedures, practices of DevOps, as well as the prescriptive steps.

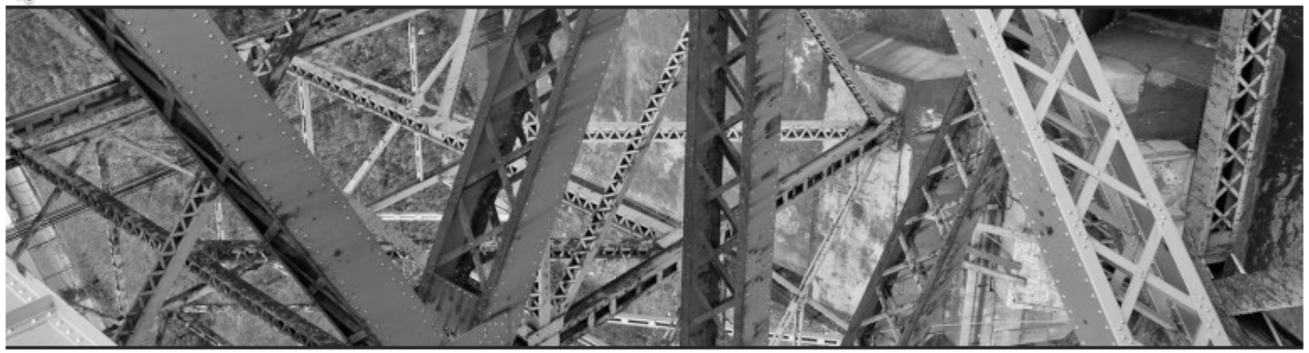
Gene Kim



Immutable Delivery

the agile admin

HOME ABOUT WHAT IS DEVOPS?



← Container Automation: Building a Brickyard

Heterogeneous Hardware Management and Benchmarking via Docker →

BY BOTCHAGALUPE | NOVEMBER 24, 2015 · 10:00 AM

↓ Jump to Comments

Immutable Delivery

This article proposes a design pattern modeled after "Immutable Infrastructure", something I call "Immutable Delivery". There has been a lot of debate and discussion about the usage of the term "Immutable" lately. Let me clearly say that there is no such thing as an immutable server or immutable infrastructure. I know of no example in my 35 years of working with IT infrastructure of a system or infrastructure that is completely immutable. A system changes and continues

Search

Subscribe

Enter your email address to subscribe to the Agile Admin and receive notifications of new posts by email.

Join 2,739 other followers

Enter your email address

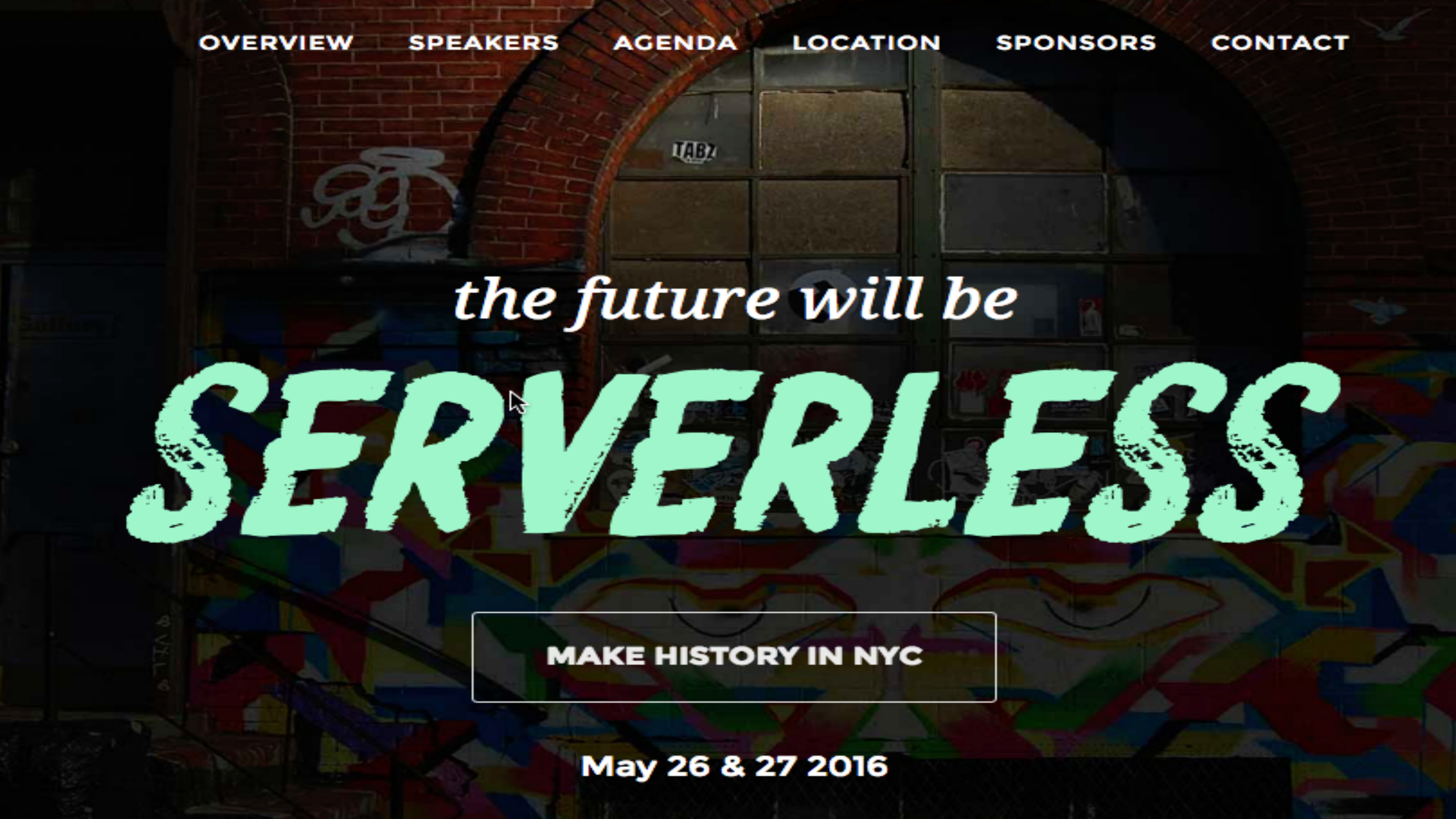
Sign me up!

Recent Comments

- The Present and Future of Configuration Management | Nordic

Immutable Infrastructure

"This is how we run our infrastructure. One of the things that developers have to do is provide the commands to start the Docker container, and that's it. This is kind of amazing right? Any EC2 instance that we spin up now, we don't care if you're running Node, Ruby, Scala, Java or if you made up your own programming language. It's absolutely amazing how nice this is. When we compare this to the way we did this in the past, we had one repository that had all of the different scripts to know how to build all of the different applications at Gilt. We have 1000 Git repos and over 300 different applications. We are 7 years old which means we have like 7 different ways of producing applications. There's 25 different ways that we build software at Gilt and it's all captured in a central repo. That was at conflict with where we are going in terms of teams really owning their software and being able to deploy their own services."

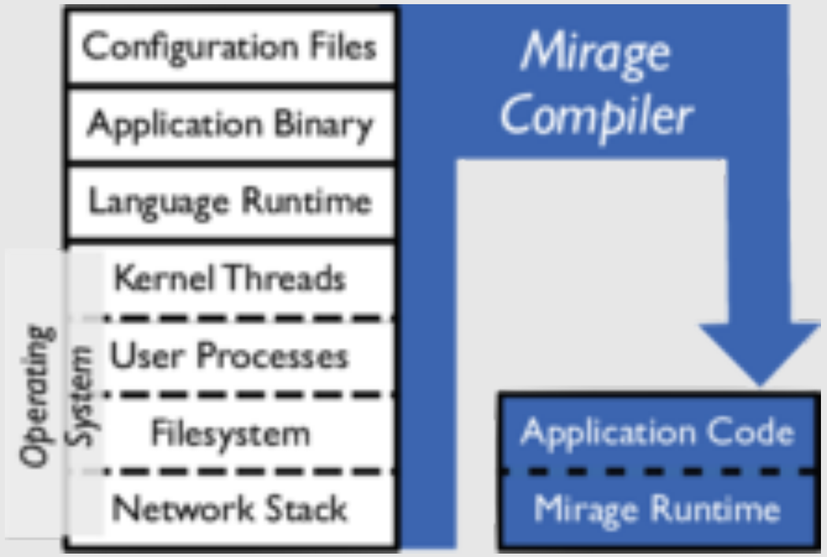

OVERVIEW
SPEAKERS
AGENDA
LOCATION
SPONSORS
CONTACT
the future will be
SERVERLESS
MAKE HISTORY IN NYC
May 26 & 27 2016

Serverless

- **AWS Lambda**
- **Azure Functions**
- **Google Cloud Functions**
- **Unikernels**

Enter Unikernels

Unikernels are specialized virtual machine images complied from the modular stack of application code, system libraries and configuration.



Enter Unikernels

docker

Docs Support Training Tech Blog

Why Docker? Products Community Partners

Docker Blog

Categories: **General** **Engineering** **Community**

January 21, 2016

Unikernel Systems Joins Docker

By **Mano Marks** - Posted in Docker, News - Tagged with docker, unikernel, unikernel system, unikernels

I'm happy to announce today that Unikernel Systems is part of Docker!

Unikernels compile your source code into a custom operating system that includes only the functionality required by the application logic. That makes them small, fast, and improves efficiency. Unikernel Systems was formed last year to build tools that allow developers to take advantage of a growing number of unikernel projects.

Unikernels

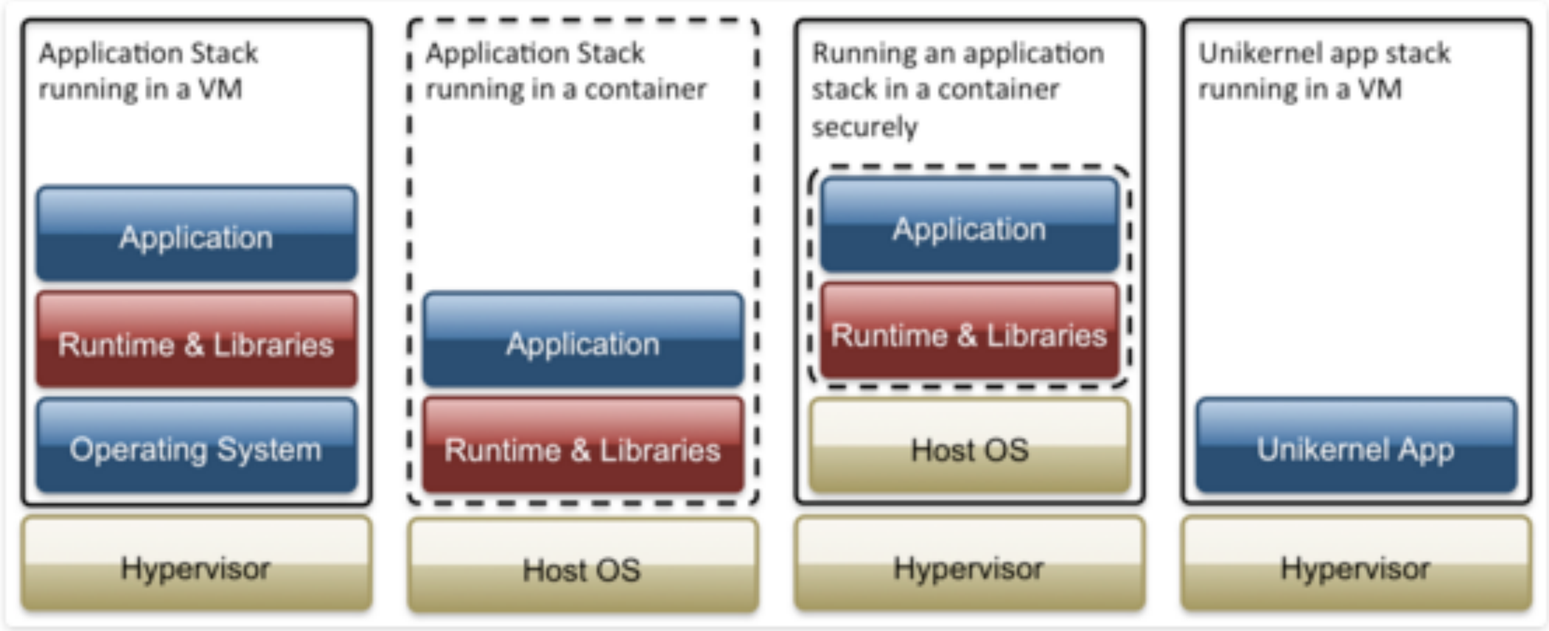


Image credit: Xen Project.

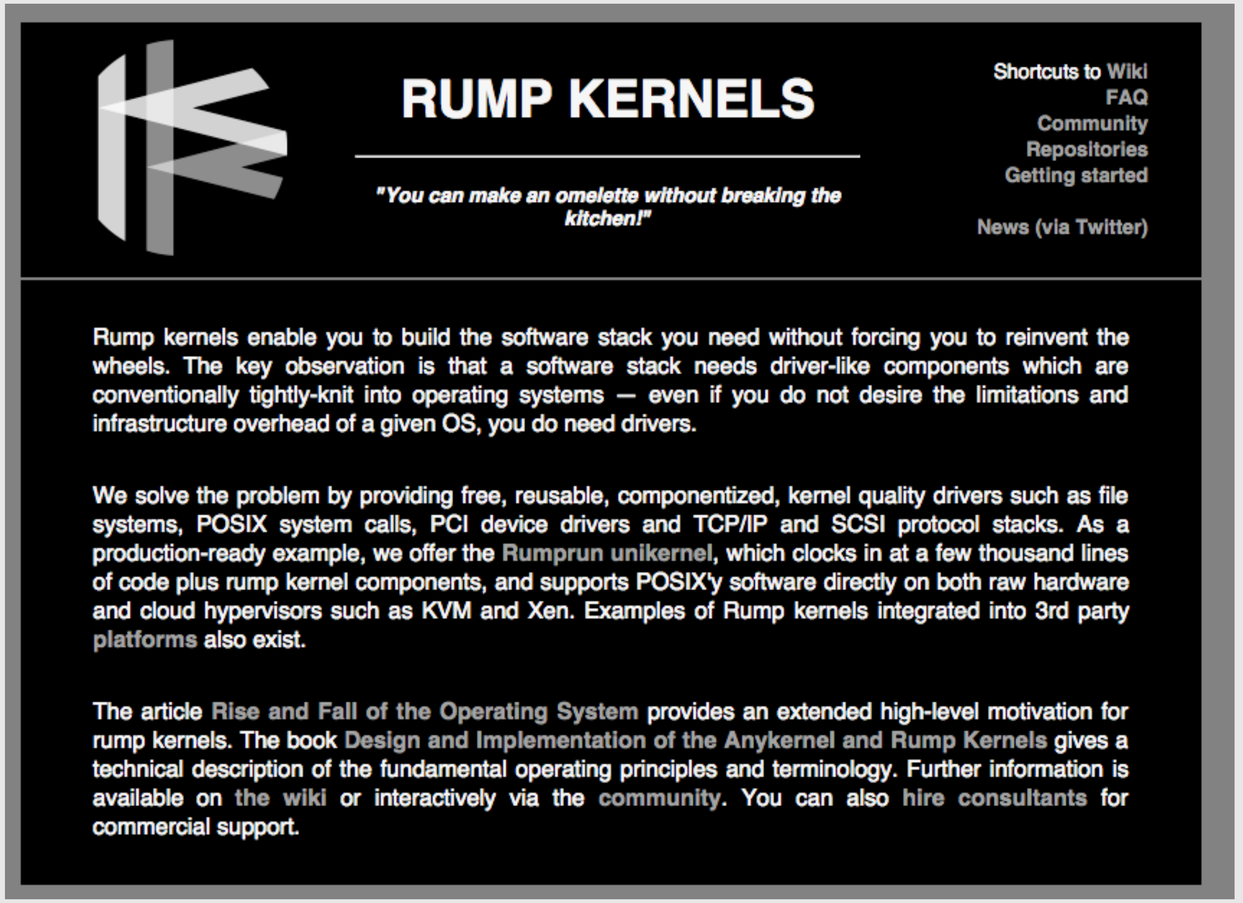
Unikernels

TABLE 2 **Other unikernel implementations**

Unikernel	Language	Targets
Mirage[13]	OCaml	Xen, kFreeBSD, POSIX, WWW/js
Drawbridge[16]	C	Windows "picoprocess"
HalVM[8]	Haskell	Xen
ErlangOnXen	Erlang	Xen
OSv[2]	C/Java	Xen, KVM
GUK	Java	Xen
NetBSD "rump"[9]	C	Xen, Linux kernel, POSIX
ClickOS [14]	C++	Xen

https://queue.acm.org/detail.cfm?id=2566628

Unikernels



RUMP KERNELS

"You can make an omelette without breaking the kitchen!"

Shortcuts to Wiki
FAQ
Community
Repositories
Getting started

News (via Twitter)

Rump kernels enable you to build the software stack you need without forcing you to reinvent the wheels. The key observation is that a software stack needs driver-like components which are conventionally tightly-knit into operating systems — even if you do not desire the limitations and infrastructure overhead of a given OS, you do need drivers.

We solve the problem by providing free, reusable, componentized, kernel quality drivers such as file systems, POSIX system calls, PCI device drivers and TCP/IP and SCSI protocol stacks. As a production-ready example, we offer the Rumprun unikernel, which clocks in at a few thousand lines of code plus rump kernel components, and supports POSIX'y software directly on both raw hardware and cloud hypervisors such as KVM and Xen. Examples of Rump kernels integrated into 3rd party platforms also exist.

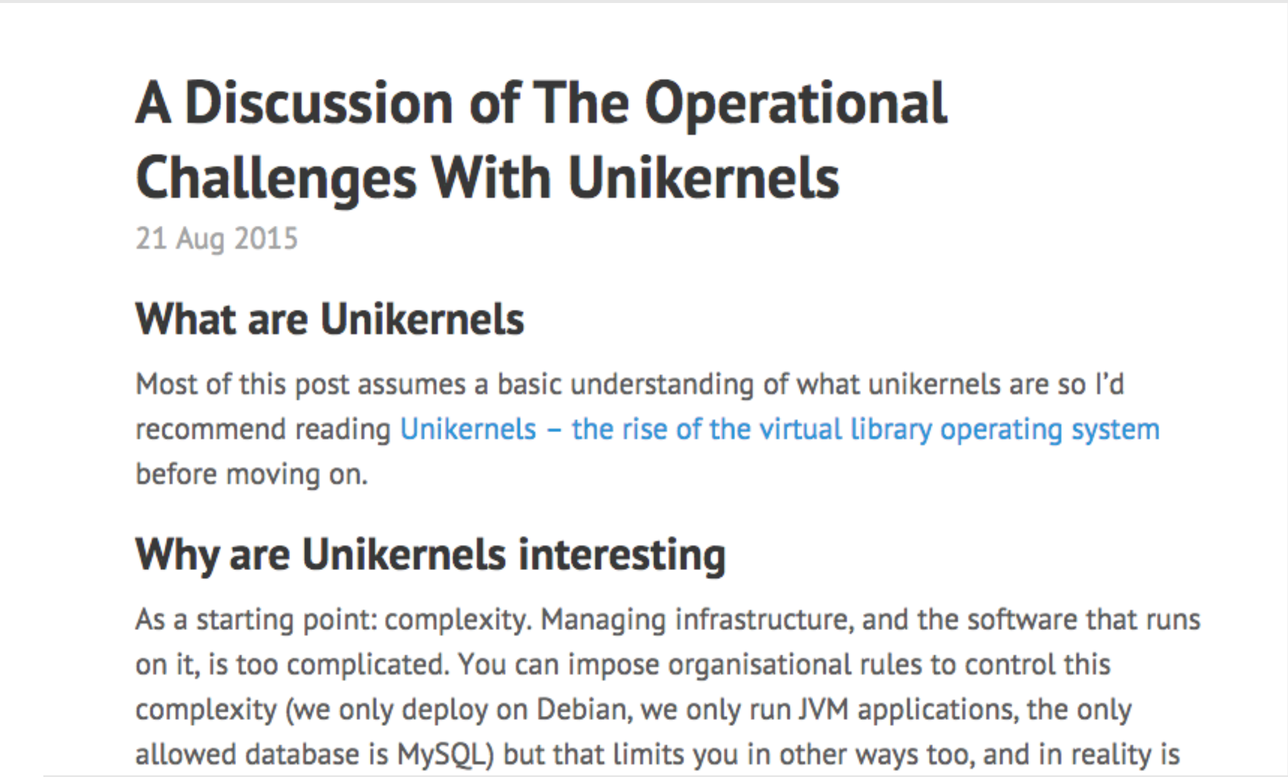
The article Rise and Fall of the Operating System provides an extended high-level motivation for rump kernels. The book Design and Implementation of the Anykernel and Rump Kernels gives a technical description of the fundamental operating principles and terminology. Further information is available on the wiki or interactively via the community. You can also hire consultants for commercial support.

http://rumpkernel.org/

Why Unikernels

- **Performance**
  - user-kernel context switches
  - instantiation times
  - Memory footprint
- **Security**
  - less attack surface
  - No known architecture patterns
- **Fine-grained optimisation**
  - as unikernels are constructed through a coherent compiler tool-chain, whole-system optimisation can be carried out across device drivers and application logic, potentially improving specialisation further

Enter Unikernels



A Discussion of The Operational Challenges With Unikernels

21 Aug 2015

What are Unikernels

Most of this post assumes a basic understanding of what unikernels are so I'd recommend reading Unikernels – the rise of the virtual library operating system before moving on.

Why are Unikernels interesting

As a starting point: complexity. Managing infrastructure, and the software that runs on it, is too complicated. You can impose organisational rules to control this complexity (we only deploy on Debian, we only run JVM applications, the only allowed database is MySQL) but that limits you in other ways too, and in reality is

Part of this is a numbers game – to run a reasonable system you might need to run 50 different services, and install 200 packages on every host. An attacker has to compromise just one of those to win - Gareth Rushgrove

Unikernel Examples

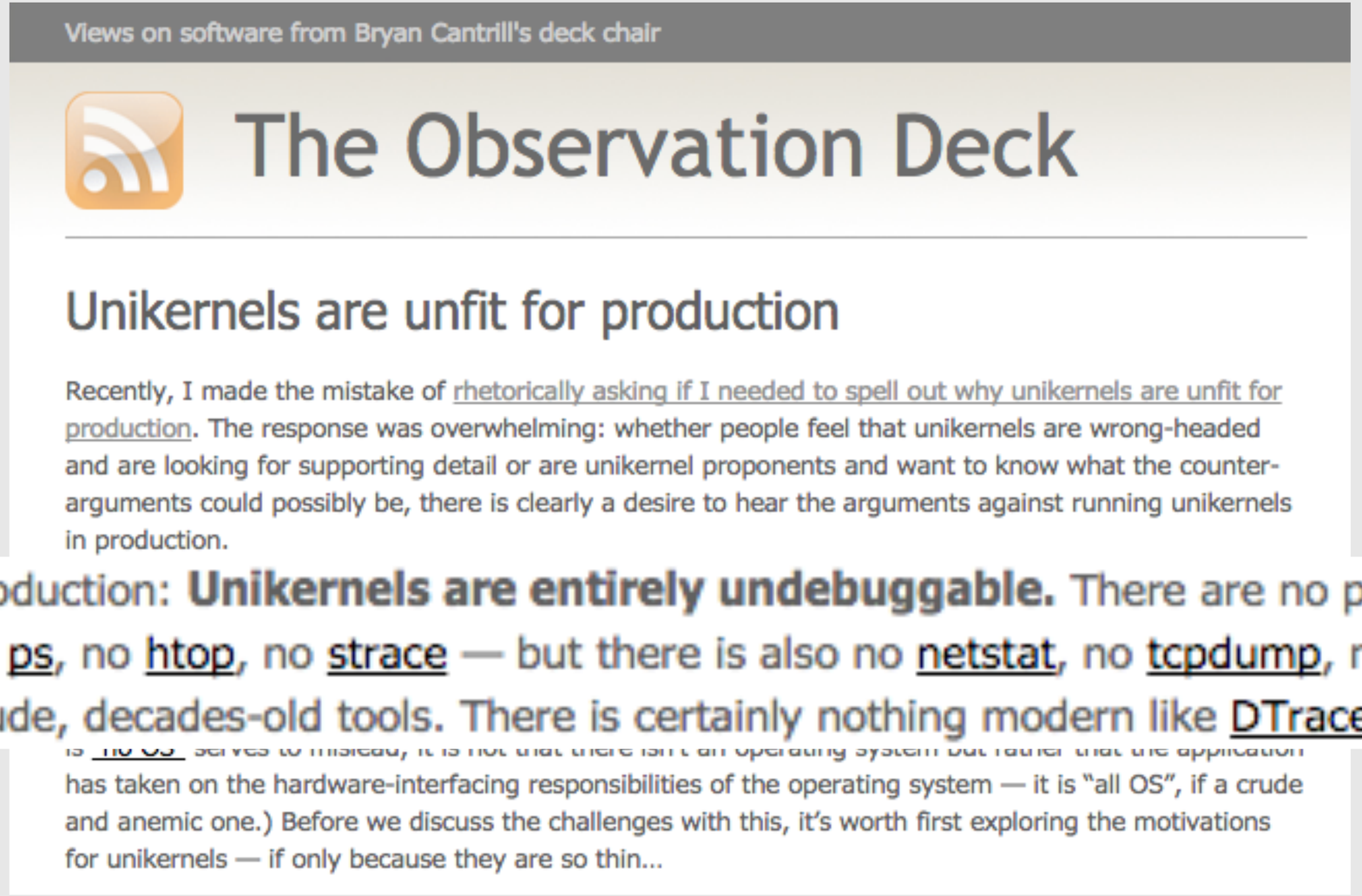
- **DNS Server 446 KB**
- **Web Server 674 KB**
- **OVS Switch 393 KB**

- **NTP server un-hacked for over a year**
- **Docker for Mac/Windows**

Unikernel Opportunities

- **Composition and Orchestration**
- **Logging and Monitoring**
- **Networking**
- **Debugging**
- **Forces Immutability**

Unikernels



Views on software from Bryan Cantrill's deck chair

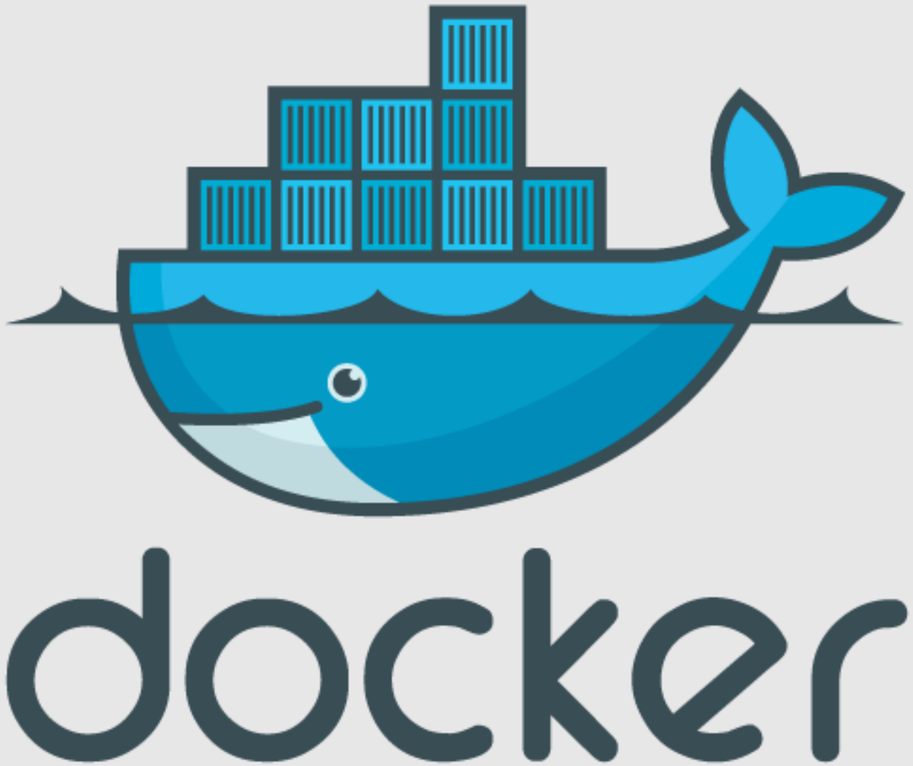
The Observation Deck

Unikernels are unfit for production

Recently, I made the mistake of rhetorically asking if I needed to spell out why unikernels are unfit for production. The response was overwhelming: whether people feel that unikernels are wrong-headed and are looking for supporting detail or are unikernel proponents and want to know what the counter-arguments could possibly be, there is clearly a desire to hear the arguments against running unikernels in production.

production: **Unikernels are entirely undebuggable.** There are no processes, so of course there is no ps, no htop, no strace — but there is also no netstat, no tcpdump, no ping! And these are just the crude, decades-old tools. There is certainly nothing modern like DTrace or MDB. From a debugging

has taken on the hardware-interfacing responsibilities of the operating system — it is "all OS", if a crude and anemic one.) Before we discuss the challenges with this, it's worth first exploring the motivations for unikernels — if only because they are so thin...



john.willis@docker.com
@botchagalupe

http://ow.ly/Xt2ro